JNWBitSet Description
Version 1.000

JNWBitSet is a class of methods to support bit processing in Java Network Workbench. JNWBitSet is a subclass of java.util.BitSet.

java.util.BitSet represents bits as a vector of boolean values (true and false). It is a very flexible class since it allows bit sets to grow to any size. It has many methods to support bit sets. You can find a JavaDoc for java.util.BitSet at http://java.sun.com/j2se/1.5.0/docs/api/java/util/BitSet.html .

Unfortunately, java.util.BitSet has a problem from our point of view. In java.util.BitSet the bit set 00000000 00000000 has length 0. java.util.BitSet uses the position of the first 1 as its length. We want 00000000 00000000 to have length 16.

Therefore, it was necessary to create the subclass Utility.JNWBitSet which corrects this error. The correct length, for our purposes, is in JNWLength. Be very careful when using any of the methods of class java.util.BitSet since they may produce unexpected results.

There are several things you should know about JNWBitSet.

java.util.BitSet and Utility.JNWBitSet represent bits as booleans (trues and falses) not 0s and 1s. So, if you went to set the indexValue bit to 0, you must use

```
set (indexValue, false);
```

There are two methods in Utility.JNWBitSet, getBitValue and setBitValue, which get and set bits as 0s and 1s. They internally convert 0s and 1s to falses and trues.

A Utility.JNWBitSet with length JNWLength has an index from 0 to JNWLength-1 as shown in the figure below

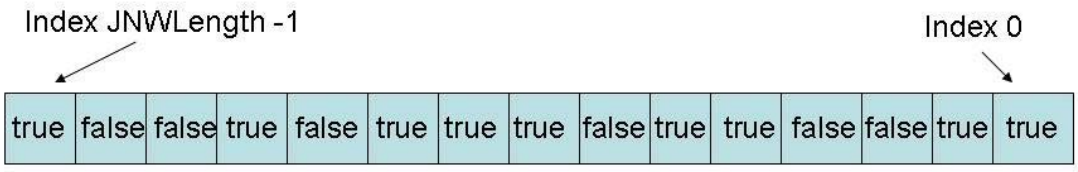

Figure 1 Indices of a JNWBitSet

JNWBitSet has several useful methods

- setJNWLength (int length) set the length of the JNWBitSet
- getJNWLength () gets the length of the JNWBitSet
- setBitValue (int bitindex , int value) set the bit at index bitIndex to 0 or 1
- getBitValue (int bitindex) gets the bit at index bitIndex as a 0 or 1
- shift (int numberOfBitsToShift) shifts the BitSet left (positive) or right (negative) by numberOfBitsToShift bits
- setJNWBitSetToString ( ) converts a character String into a JNWBitSet
- writeBitSet () converts a JNWBitSet to a String of 0s and 1s

Notice that a shift with a positive index causes the bits to shift to the left and a shift with a negative shift causes the bits to shift to the right.

There are several methods to convert between Strings, integers, etc. and JNWBitSets. They include

- setJNWBitSetToLongAndTruncate (long inputIn , int numberOfBits) converts a long, int, short, or byte to a bit set and truncates it to a specified number of bits
- setJNWBitSetToString (String stringOfCharacters) converts a String to a bit set
- setJNWBitSetToZerosAndOnes (String stringOfZerosAndOnes) converts a String of 0s and 1s, such as “01010011 10101010” to a bit set
- convertJMUBitSetToLong ( ) converts a bit set to a long
- convertJMUBitSetToString ( ) converts a bit set to a String

There are several methods for appending and prepending bit sets. For example,

- data.append (“01010011 10101010”) will append “01010011 10101010” to the end of data
- data.prepend (“01010011 10101010”) will append “01010011 10101010” as the header of data

Utility.JNWBitSet also has several useful constructors

- JNWBitSet(int nbits) creates a JNWBitSet of length nbits with all the bits false (0)

- JNWBitSet(JNWBitSet JNWBitSetIn) is a cloning constructor which creates an identical copy of JNWBitSetIn. The clone () method does not work correctly with java.util.BitSet
- JNWBitSet(String stringOfZerosAndOnes) creates a bit set of 0s and 1s. For example,

  JNWBitSet data = new JNWBitSet (“00001111 1111 0000 10”) ;

  creates the bit set 000011111111000010.